Gestão e Qualidade de Software

Integrantes do grupo:

Kayky Cerquiaro Prado - 822155538
Enrico Aguiar Vrunski - 82210618
Thiago Ferreira Lima Gonçalves - 824156179
Matheus Tognon Siqueira - 824156311
Felipe Soares de Oliveira - 824156311

São Paulo
2024

O Dilema da Qualidade de Software envolve encontrar um equilíbrio entre qualidade, custo e tempo de desenvolvimento.

1. Software “Bom o Suficiente”

Esse conceito parte da ideia de que um software não precisa ser perfeito para ser útil e viável. Em muitos casos, empresas lançam versões "boas o suficiente" para atender às necessidades dos usuários e ao mesmo tempo manter um custo e tempo de desenvolvimento razoáveis.

Trade-offs: Buscar a perfeição pode atrasar lançamentos e aumentar os custos, enquanto lançar algo inacabado pode gerar frustração e perda de clientes.

Casos comuns: Softwares SaaS (Software como Serviço) muitas vezes são lançados com o mínimo de funcionalidades e depois aprimorados conforme o feedback dos usuários (MVP - Produto Mínimo Viável).

Exemplo: Um aplicativo de e-commerce pode ser lançado sem um sistema de recomendação de produtos avançado, mas com funcionalidades essenciais como catálogo e pagamento seguro.

2. Custo da Qualidade

O custo da qualidade refere-se ao investimento necessário para garantir que o software atenda aos requisitos esperados, além dos custos associados a falhas. Ele se divide em quatro categorias:

1. Custos de Prevenção – Investimentos em processos, treinamentos e boas práticas para evitar defeitos antes que aconteçam.
2. Custos de Avaliação – Testes, revisões de código e auditorias para garantir a qualidade do software antes do lançamento.
3. Custos de Falhas Internas – Correção de erros antes do software chegar ao cliente (retrabalho, depuração).
4. Custos de Falhas Externas – Problemas identificados após a entrega, que podem resultar em suporte técnico, perda de reputação e até ações judiciais.

Equilíbrio necessário: Empresas precisam investir nos primeiros itens (prevenção e avaliação) para evitar gastos ainda maiores com falhas externas.

Exemplo: Se uma fintech não testa adequadamente seu aplicativo, pode ter um erro crítico em transações financeiras, gerando prejuízos e perda de credibilidade.

Portanto, o dilema entre desenvolver um software "bom o suficiente" e investir no custo da qualidade é um desafio estratégico para qualquer empresa de tecnologia. Encontrar um equilíbrio entre qualidade, prazo e custo é essencial para o sucesso do software.

Negligência e Responsabilidade Social

É um dilema importante quando se desenvolve o projeto pois quando está sendo criado o software é importante que seja aplicado as melhores práticas de engenharia, desenvolvimento, testes e segurança para que o software seja entregue com qualidade, desempenho e segurança. Caso as práticas de software não sejam aplicadas da melhor maneira entra a questão da negligência na criação e manutenção de software de qualidade o que pode gerar consequências desastrosas para indivíduos e isso se relaciona com a responsabilidade social pois se o software não for entregue com qualidade e bom desempenho pode afetar diretamente indivíduos, órgãos de serviços públicos e empresas.

Riscos

- Baixa Qualidade

Esse dilema é muito importante quando estamos realizando um projeto pois a baixa qualidade aumenta os riscos tanto para o desenvolvedor quanto para o usuário. Pois entregar um produto com baixa qualidade pode trazer mais falhas e pode gerar custos

- Pressão por Prazo

Esse é um dilema que pode ser enfrentado onde muitas vezes há a demanda para que o desenvolvimento do sistema seja entregue em prazos mais curtos, isso pode fazer com que a equipe reduza testes e revisões com objetivo de atender ao prazo, o que pode estar relacionado diretamente com os riscos pois com menos testes e revisões pode trazer um produto final com mais falhas.

- Atender ao orçamento

No desenvolvimento do Software pode aparecer um dilema relacionado ao orçamento, o projeto tem que estar dentro do orçamento planejado. O que pode estar também relacionado com os riscos já que os testes necessários custam e pode acabar se tornando caro, o que pode trazer uma redução de testes para não comprometer o orçamento.

- Necessidades de equilíbrio

Esse é outro dilema que pode ser enfrentado no desenvolvimento do projeto pois se o software é produzido e entregue com uma qualidade mais baixa a empresa perde pois ninguém ira quer usá-lo, mas também tem o outro ponto que se é gasto um tempo muito grande, um esforço extremamente grande e quantidades altas de dinheiro para construir um software absolutamente perfeito faz com que ele leve muito tempo para ser concluído, e o custo de produção será tão alto o que pode trazer problemas para a empresa. Então essa necessidade de buscar equilíbrio pode ser um dilema pois precisamos aproveitar as oportunidades de mercado, mas sem esgotar todos os recursos da empresa e entregar um produto bom o suficiente sem levar um tempo e custos muito absurdos.

Qualidade e Segurança
Conceito Amplo:

A qualidade do software é um tema central no desenvolvimento tecnológico. Garantir que um sistema atenda às necessidades dos usuários e aos requisitos definidos não é uma tarefa simples, pois envolve equilibrar agilidade e robustez. Qualidade de software pode ser definida como a capacidade de um sistema desempenhar suas funções de maneira correta, eficiente e segura. Para isso, consideram-se aspectos como:

- Funcionalidade: o software realiza corretamente as funções para as quais foi projetado?
- Confiabilidade: o sistema opera sem falhas, mesmo sob condições adversas?
- Usabilidade: a interface é intuitiva e de fácil compreensão para o usuário?
- Eficiência: há um uso otimizado de recursos computacionais?
- Segurança: os dados e operações estão protegidos contra acessos não autorizados?

Papel Principal:
A segurança é um pilar essencial da qualidade de software. Um sistema rápido e funcional perde seu valor se expõe dados sensíveis ou permite ataques. A segurança não é apenas uma camada adicional, mas uma base que sustenta toda a estrutura do software. Para garantir um sistema seguro, consideram-se os seguintes princípios:

- Confidencialidade: proteção de informações privadas contra acessos indevidos, garantindo que apenas usuários autorizados possam visualizar ou manipular dados sensíveis.
- Integridade: prevenção contra modificações não autorizadas nos dados, assegurando que as informações permaneçam corretas e consistentes durante todo o ciclo de vida do software.
- Disponibilidade: garantia de que o sistema estará acessível sempre que necessário, evitando interrupções que possam afetar o funcionamento ou a experiência do usuário.

Agilidade ou Confiança?

Esse dilema se intensifica com metodologias ágeis, que promovem entregas incrementais e rápidas. Embora essa abordagem acelere a inovação, ela pode, sem as devidas precauções, resultar em código mal testado ou práticas de segurança negligenciadas. Por outro lado, investir tempo excessivo em revisões e testes rigorosos pode fazer com que o produto perca o timing de mercado, resultando em desvantagem competitiva, por isso é importante equilibrar estes dois objetivos muitas vezes conflitantes:

- Entrega rápida: a pressão do mercado exige que novos produtos e funcionalidades sejam lançados o quanto antes.
- Robustez e segurança: testar, revisar e garantir a solidez do software demanda tempo e recursos.

A qualidade do software é um fator fundamental para o sucesso de qualquer projeto de desenvolvimento, e seu impacto vai além de aspectos técnicos, abrangendo gestão, planejamento e cultura organizacional. Para alcançar um alto padrão de qualidade, é essencial que a administração adote práticas e estratégias que garantam um processo eficiente e seguro, evitando riscos e falhas ao longo do ciclo de vida do software.

Um dos pilares para garantir a qualidade é o planejamento e o gerenciamento de projetos. A definição clara de requisitos e objetivos desde o início assegura que o software entregue atenda

às expectativas dos usuários e cumpra as demandas do mercado. A escolha da metodologia de desenvolvimento, seja ágil ou tradicional, também influencia diretamente na eficiência do processo e na capacidade de adaptação a mudanças. Além disso, a gestão adequada de riscos contribui para mitigar falhas e evita retrabalho, promovendo um desenvolvimento mais robusto e sustentável.

A alocação de recursos também desempenha um papel crucial na busca pela qualidade. Contar com equipes bem estruturadas e compostas por profissionais qualificados garante um desenvolvimento mais eficiente e assertivo. A distribuição equilibrada de tarefas evita a sobrecarga dos colaboradores e contribui para a produtividade, enquanto investimentos em ferramentas e infraestrutura adequadas aumentam a confiabilidade e a estabilidade do software, proporcionando um ambiente propício para a criação de soluções inovadoras e seguras.

Outro aspecto fundamental é a definição de processos e padrões que norteiem o desenvolvimento. Estabelecer padrões de codificação promove a legibilidade e a manutenibilidade do código, facilitando futuras atualizações e correções. Além disso, a adoção de boas práticas, como revisões de código e integração contínua, reduz a incidência de erros e melhora a qualidade geral do produto. O cumprimento de normas e certificações também contribui para garantir a conformidade e a segurança do software, fortalecendo a confiança dos usuários e clientes.

A gestão da qualidade e a realização de testes são etapas indispensáveis para garantir um produto confiável e seguro. A implementação de testes automatizados e manuais permite identificar falhas antes que elas alcancem o ambiente de produção, enquanto o monitoramento contínuo e o feedback dos usuários ajudam a corrigir problemas rapidamente. Auditorias e revisões periódicas garantem que o software continue atendendo aos requisitos, mesmo após atualizações e modificações.

Por fim, a cultura organizacional e a comunicação entre equipes desempenham um papel vital na promoção da qualidade. Uma cultura que valoriza boas práticas e incentiva o compromisso com a excelência contribui para um ambiente de trabalho colaborativo e produtivo. Além disso, uma comunicação clara e eficaz entre desenvolvedores, testadores e gestores reduz falhas e retrabalho, garantindo que todos os envolvidos estejam alinhados em relação aos objetivos e às expectativas do projeto.

Em suma, a busca pela qualidade no desenvolvimento de software requer um esforço conjunto que envolve planejamento estratégico, gestão de recursos, definição de processos robustos, práticas de testes eficientes e uma cultura organizacional comprometida com a excelência. Somente assim é possível garantir soluções tecnológicas seguras, confiáveis e capazes de atender às demandas do mercado com qualidade e eficiência.

Bibliografia

PRESSMAN, Roger; MAXIM, Bruce. Engenharia de Software. Uma abordagem profissional. 9a. Ed. Bookman, 2021.
Parte III – Gestão da qualidade; Capítulo 15 – Conceitos de qualidade, pg. 310
https://integrada.minhabiblioteca.com.br/reader/books/9786558040118/epubcfi/6/2[%3Bvnd.vst.idref%3DCapa.xhtml]!/4/2[page_i]/2%4051:1

SOMMERVILLE, Ian. Engenharia de Software. 9. ed. São Paulo: Pearson Prentice Hall, 2011.
https://bv4.digitalpages.com.br/?term=engenharia%2520de%2520software&searchpage=1&filtro=todos&from=busca&page=_14&section=0#/legacy/276

https://owasp.org/www-project-top-ten/